We Claim:

1. A method of identifying a global breakpoint for debugging computer software, said method including the steps of:

representing said global breakpoint in code of said software using an identifier of an executable file and an offset in said executable file.

2. The method according to claim 1, wherein said file identifier is a file name.

3. The method according to claim 1, wherein said file identifier is an inode of a Unix or Unix-like operating system.

4. The method according to claim 1, wherein said file identifier is a file control block of a non-Unix operating system.

5. The method according to claim 1, further including the step of resolving a virtual address of said code to said file identifier and said offset.

6. The method according to claim 1, wherein said offset is obtained using symbol expressions.

7. The method according to claim 1, further including the step of providing a hash list to look up said global breakpoint using said file identifier and said offset.

8. The method according to claim 7, further including the step of maintaining said hash list of global breakpoints based on file identifiers and offsets.

9. The method according to claim 1, further including the step of deriving said file identifier and said offset using a virtual address.

10. The method according to claim 9, wherein said deriving step is dependent upon information maintained by an operating system to map executable files to memory.

11. The method according to claim 9, further including the step of:

determining file identifier and said offset from said virtual address for a memory mapped region.

12. The method according to claim 9, wherein two or more virtual addresses exist for said software code.

13. The method according to claim 1, wherein said global breakpoint is contained in a private-per-process copy of a physical page of said software code.

14. A computer-implemented apparatus for identifying a global breakpoint for debugging computer software, said apparatus including:

a central processing unit for executing said computer software;

memory for storing at least a portion of said computer software; and

means for representing said global breakpoint in code of said computer software using an identifier of an executable file and an offset in said executable file.

15. The apparatus according to claim 14, wherein said file identifier is a file name.

16. The apparatus according to claim 14, wherein said file identifier is an inode of a Unix or Unix-like operating system.

17. The apparatus according to claim 14, wherein said file identifier is a file control block of a non-Unix operating system.

18. The apparatus according to claim 14, further including means for resolving a virtual address of said code to said file identifier and said offset.

19. The apparatus according to claim 14, wherein said offset is obtained using symbol expressions.

20. The apparatus according to claim 14, further including a hash list to look up said global breakpoint using said file identifier and said offset.

21. The apparatus according to claim 20, further including means for maintaining said hash list of global breakpoints based on file identifiers and offsets.

22. The apparatus according to claim 14, further including means for deriving said file identifier and said offset using a virtual address.

23. The apparatus according to claim 22, wherein said deriving means is dependent upon information maintained by an operating system to map executable files to memory.

24. The apparatus according to claim 22, further including means for determining said file identifier and said offset from said virtual memory address for a memory mapped region.

25. The apparatus according to claim 22, wherein two or more virtual addresses exist for said software code.

26. The apparatus according to claim 14, wherein said global breakpoint is contained in a private-per-process copy of a physical page of said software code.

27. A computer program product having a computer readable medium having a computer program recorded therein for identifying a global breakpoint for debugging computer software, said computer program product including:

computer program code means for representing said global breakpoint in code of said computer software using an identifier of an executable file and an offset in said executable file.

28. The computer program product according to claim 27, wherein said file identifier is a file name.

29. The computer program product according to claim 27, wherein said file identifier is an inode of a Unix or Unix-like operating system.

30. The computer program product according to claim 27, wherein said file identifier is a file control block of a non-Unix operating system.

31. The computer program product according to claim 27, further including computer program code means for resolving a virtual address of said code to said file identifier and said offset.

32. The computer program product according to claim 27, wherein said offset is obtained using symbol expressions.

33. The computer program product according to claim 27, further including computer program code means for providing a hash list to look up said global breakpoint using said file identifier and said offset.

34. The computer program product according to claim 33, further including computer program code means for maintaining said hash list of global breakpoints based on file identifiers and offsets.

35. The computer program product according to claim 27, further including computer program code means for deriving said file identifier and said offset using a virtual address.

36. The computer program product according to claim 35, wherein said computer program code means for deriving is dependent upon information maintained by an operating

system to map executable files to memory.

37. The computer program product according to claim 35, further including computer program code means for determining said file identifier and said offset from said virtual memory address for a memory mapped region.

38. The computer program product according to claim 35, wherein two or more virtual addresses exist for said software code.

39. The computer program product according to claim 27, wherein said global breakpoint is contained in a private-per-process copy of a physical page of said software code.